Requiem for local single–pass methods solving stationary Hamilton–Jacobi equations?*

Simone Cacace†, Emiliano Cristiani‡, Maurizio Falcone§

January 30, 2013

Abstract

The use of *local single-pass* methods (like, e.g., the Fast Marching method) has become popular in the solution of some Hamilton-Jacobi equations. The prototype of these equations is the *eikonal equation*, for which the methods can be applied saving CPU time and possibly memory allocation. Then, some natural questions arise: can local single-pass methods solve every Hamilton-Jacobi equation? If not, where the limit should be set?

This paper tries to answer these questions. In order to give a complete picture, we present an overview of some fast methods available in literature and we briefly analyze their main features. We also introduce some numerical tools and provide several numerical tests which are intended to exhibit the limitations of the methods. We show that the construction of a local single-pass method for general Hamilton-Jacobi equations is very hard, if not impossible. Nevertheless, some special classes of problems can be actually solved, making local single-pass methods very useful from the practical point of view.

Keywords. Fast Marching methods, Fast Sweeping methods, eikonal equation, Hamilton-Jacobi equations, semi-Lagrangian schemes.

AMS. 65N12, 49L20, 49M99

1 Introduction

The study of Hamilton-Jacobi (HJ) equations arises in several applied contexts, including classical mechanics, front propagation, control problems and differ-

*The authors wish to acknowledge the support obtained by the following grants: AFOSR Grant no. FA9550-10-1-0029, ITN - Marie Curie Grant no. 264735-SADCO and SAPIENZA 2009 "Analisi ed approssimazione di modelli differenziali nonlineari in fluidodinamica e scienza dei materiali".

†Dipartimento di Matematica, SAPIENZA – Università di Roma, P.le Aldo Moro 2, 00185 Rome, Italy. cacace@mat.uniroma1.it

‡Istituto per le Applicazioni del Calcolo, Consiglio Nazionale delle Ricerche, Rome, Italy. e.cristiani@iac.cnr.it

§(corresponding author) Dipartimento di Matematica, SAPIENZA – Università di Roma, P.le Aldo Moro 2, 00185 Rome, Italy. falcone@mat.uniroma1.it

ential games, and it has a great impact in many areas, such as robotics, aeronautics, electrical and aerospace engineering. In particular, for control/game problems, an approximation of the value function allows for the synthesis of optimal control laws in feedback form, and then for the computation of optimal trajectories. The value function for a control problem (resp. differential game) can be characterized as the unique viscosity solution of the corresponding Hamilton-Jacobi-Bellman (HJB) equation (resp. Hamilton-Jacobi-Isaacs (HJI) equation), and it is obtained by passing to the limit in the well known Bellman's Dynamic Programming (DP) principle. The DP approach can be rather expensive from the computational point of view, but it gives a real advantage when compared to methods based on the Pontryagin's Maximum Principle, because the latter approach allows one to compute only open-loop controls and sub-optimal trajectories. Moreover, weak solutions to HJ equations are nowadays well understood in the framework of viscosity solutions, which offers the correct notion of solution for many applied problems. The above remarks have motivated the research of efficient and accurate numerical methods. Indeed, an increasing number of techniques have been proposed for the approximation of viscosity solutions. They range from Finite Difference to Finite Volume, from Discontinuous Galerkin to semi-Lagrangian schemes. In any case, for optimal control problems and games, the DP approach suffers from the so-called "curse of dimensionality" limitation, i.e. the size of nonlinear systems to solve for high dimensional problems becomes huge and this makes the numerical solution unfeasible, in terms of both memory allocation and CPU time. The curse of dimensionality can be overcome only by exploiting the peculiarities of every single problem (e.g., symmetry, periodicity, linearity) or by adopting a linearization based on the so called "max-plus algebra" approach, which unfortunately presents other types of constraints, see, e.g., the book by McEneaney [18]. It is rather clear that the DP approach needs a big effort in the construction of numerical approximation schemes for two different reasons. The first, which is valid even in low dimension, is due to the low regularity of viscosity solutions which are typically only Lipschitz continuous and (in some cases such as constrained control problems and pursuit-evasion games) can also be discontinuous. The second reason is related to the above mentioned curse of dimensionality which pushes towards methods with low memory allocation in the construction of the local solver and, possibly, the definition of some rule to reduce the number of elementary operations and the CPU time. Another motivating field is the approximation of front propagation problems via the level-set method. The motivation there is to reduce or eliminate the extra dimension which is added by the level-set method and obtain a fast and reliable algorithm. Starting from the 1980s, many efforts have been made to improve the efficiency of these numerical methods, a crucial step for the solution of real-world problems.

In this paper we deal with numerical methods for solving first-order nonlinear stationary HJ equations. In particular, we focus on the applicability of Fast Marching Method (FMM), introduced in the pioneering works by Tsitsiklis [24], Sethian [21], Helmsen et al. [14], and its generalizations, see, e.g.,

[1, 4, 5, 6, 7, 8, 9, 15, 17, 19, 22]. We analyze features and limitations of this kind of algorithms, aiming at understanding if it is possible to construct local single-pass methods solving every HJ equation. Then, we discuss whether or not the research on this topic should look for new future directions still based on the local single-pass idea and/or switch to other acceleration methods, such as Fast Sweeping Methods (FSM) as proposed by several authors starting from Danielsson [10], see, e.g., [16, 23, 25].

It is well known that the FMM has been developed for the stationary/evolutive eikonal equation. This explains why we decide to use as a guide line the following equations, associated to some minimum time problems with target:

$$
\begin{aligned}
|\nabla T(x)| &= 1 && \text{(homogeneous eikonal)} && (1)\\
c_1(x)|\nabla T(x)| &= 1 && \text{(nonhomogeneous eikonal)} && (2)\\
c_2\left(\nabla T/|\nabla T|\right)|\nabla T(x)| &= 1 && \text{(homogeneous anisotropic eikonal)} && (3)\\
c_3\left(x, \nabla T/|\nabla T|\right)|\nabla T(x)| &= 1 && \text{(nonhomogeneous anisotropic eikonal)} && (4)\\
\max_{a\in A}\{-f(x,a)\cdot\nabla T(x)\} &= 1 && \text{(minimum time HJB)} && (5)
\end{aligned}
$$

where $x \in \mathbb{R}^d \backslash \mathcal{T}$, c_1, c_2, c_3 are given strictly positive and Lipschitz continuous functions, f is a given vector-valued Lipschitz continuous function, $\mathcal{T} \subset \mathbb{R}^d$ is a closed nonempty target set and $A \subset \mathbb{R}^m$ is a closed nonempty set of admissible controls, for some $m \geq 1$. To simplify the presentation we will always consider the homogeneous Dirichlet condition $T = 0$ on $\mathcal{T}$ but also other boundary conditions can be applied, provided some compatibility conditions between the vectorfield f and $\partial\mathcal{T}$ hold true. Let us also note, for the readers not familiar with control applications, that (1)-(4) are particular cases of (5), with $A = B(0,1)$ and $f(x,a) = a$, $c_1(x)a$, $c_2(a)a$, $c_3(x,a)a$, respectively. Moreover, the above relation shows the equivalence between the front propagation problem described by the level set method and the minimum time problem, as one can find in [12]. To simplify the notations, we restrict the discussion to the case $d = 2$, but similar arguments are valid in any dimension (and the situation is even worst in higher dimension).

As mentioned above, in the last decades many numerical schemes and algorithms were proposed to solve the above equations. Some of these schemes are listed in the next section, together with their main properties. As it is well known, one important feature held by Fast-Marching-like methods is that the solution to the HJ equation is computed in a finite number of steps. More precisely, these methods are *single-pass*, in the sense of the following definition.

Definition 1.1 (Single-pass algorithm) *An algorithm is said to be* single-pass *if each mesh point is re-computed at most r times, where r depends only on the equation and the mesh structure, not on the number of mesh points.*

Single-pass algorithms usually divide the numerical grid in, at least, three subsets: *Accepted* (ACC) region, *Considered* ($CONS$) region and *Far* (FAR) region. Nodes in ACC are definitively computed, nodes in $CONS$ are computed but their values are not yet final and nodes in FAR are not yet computed. We also introduce the following definition.

Definition 1.2 (Local single-pass algorithm) *A single-pass algorithm is said to be* local *if the computation at any mesh point involves only the values of first neighboring nodes, the region CONS is one-cell thick and no information coming from FAR region is used.*

The paper is organized as follows: in Section 2 we summarize some of the existing methods to solve HJ equations and introduce two semi-Lagrangian numerical schemes. In Section 3, which is the core of the paper, we discuss the main features and limitations of the methods presented in Section 2 and we address the problem of extending local single-pass methods to general HJ equations. Finally, in Section 4 we present several experiments and numerical tests, in order to confirm the scenario depicted in Section 3 and compare the two schemes described in Section 2.

2 Background and general approximation schemes

Fast methods for HJ equations are usually designed to work with different local schemes, including Finite Difference and semi-Lagrangian (SL) schemes. Several results show that, in many cases on structured grids and at a reasonable cost, SL schemes provide better accuracy than other schemes (see, e.g., [9, 13]), due to their ability to follow directions which are oblique with respect to the coordinate axes. In this section we recall, for readers convenience, two SL schemes for HJ equations, which will be compared in Section 4. Then, we list and discuss some of the iterative and Fast-Marching-like methods available in literature.

2.1 Two SL schemes

Let us introduce a structured grid G and denote its nodes by x_i, $i = 1, \dots, N$. The space step is assumed to be uniform and equal to $\Delta x > 0$. The HJ equation can be discretized by means of the discrete version of the Dynamic Programming Principle. In this way the relationship with the optimal control framework is never lost. Standard arguments [2] lead to the following discrete version of the HJB equation (5):

$$T(x_i) \approx \widehat{T}(x_i) = \min_{a \in A} \left\{ \widehat{T}(\tilde{x}_{i,a}) + \frac{|x_i - \tilde{x}_{i,a}|}{|f(x_i, a)|} \right\}, \qquad x_i \in G \tag{6}$$

where $\tilde{x}_{i,a}$ is a *non-mesh* point, obtained by integrating, until a certain final time τ, the ordinary differential equation

$$\begin{cases} \dot{y}(t) = f(y, a), & t \in [0, \tau] \\ y(0) = x_i \end{cases} \tag{7}$$

and then setting $\tilde{x}_{i,a} = y(\tau)$. To make the scheme fully discrete, the set of admissible controls A is discretized in N_c points and we denote by a^* the optimal control achieving the minimum in (6). Note that we can get different versions of the SL scheme (6) varying τ, the method used to solve (7), and the interpolation method used to compute $\widehat{T}(\tilde{x}_{i,a})$. Moreover, we remark that, in any *single-pass* method, the computation of $\widehat{T}(x_i)$ cannot involve the value $\widehat{T}(x_i)$ itself, because this self-dependency would make the method iterative.

2.1.1 A two-point SL scheme

This scheme is used, for example, in [22] and [24]. Equation (7) is solved by an explicit forward Euler scheme until the solution intercepts the line connecting two neighbouring points $x_{i,1}$ and $x_{i,2}$ (see Fig. 1a). The value $\widehat{T}(x_i)$ is computed by a one-dimensional linear interpolation of the values $\widehat{T}(x_{i,1})$ and $\widehat{T}(x_{i,2})$ with weights $\lambda_{i,1}$ and $\lambda_{i,2}$ respectively ($\lambda_{i,1} + \lambda_{i,2} = 1$).

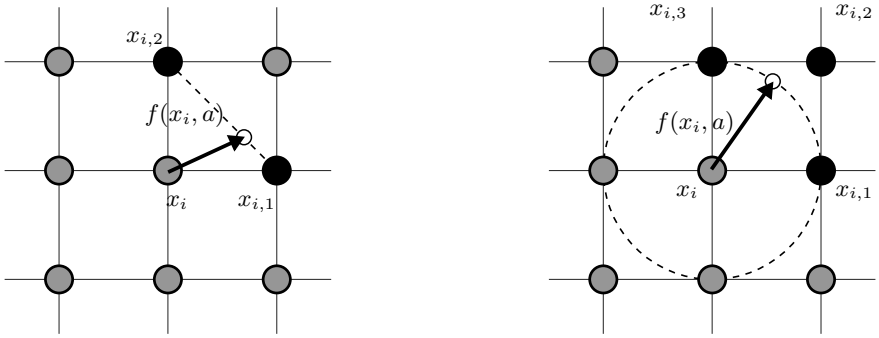


Figure 1: (a) 2-points SL scheme. (b) 3-points SL scheme

2.1.2 A three-point SL scheme

This scheme is used, for example, in [9]. Equation (7) is solved by an explicit forward Euler scheme until the solution is at distance Δx from x_i, where it falls inside the triangle of vertices $x_{i,1}$, $x_{i,2}$, and $x_{i,3}$ (see Fig. 1b). The value $\widehat{T}(x_i)$ is computed by a two-dimensional linear interpolation of the values $\widehat{T}(x_{i,1})$, $\widehat{T}(x_{i,2})$ and $\widehat{T}(x_{i,3})$ with weights $\lambda_{i,1}$, $\lambda_{i,2}$ and $\lambda_{i,3}$ respectively ($\lambda_{i,1} + \lambda_{i,2} + \lambda_{i,3} = 1$).

Remark 2.1 *It is important to note that, when coupling the schemes with Fast Marching techniques, the nodes in CONS can be either included in or excluded from the computation of* $\widehat{T}(x_i)$*. Indeed, we can decide to force the scheme to employ only nodes in ACC, temporarily assuming that nodes in CONS have very large values, so that they are automatically rejected by the minimum search in* (6)*. Otherwise, we can employ nodes in ACC and CONS, although the values at nodes in CONS are not in general correct since they can still vary in the following iterations.*

2.2 Some algorithms for HJ equations

Here we list and briefly describe some iterative and *single-pass* methods for solving HJ equations.

Iterative Method (ITM) This method is the classical one, see, e.g., [13, 20]. Starting from some initial guess $\widehat{T}^{(0)}$ defined on the whole grid (compatible with the Dirichlet conditions imposed on $\mathcal{T}$) the nodes are visited in some unique and predefined order. At each visit, the numerical scheme is applied and a new value for the node is computed. This leads to a fixed-point algorithm of the form

$$\widehat{T}^{(n)} = \widehat{H}(\widehat{T}^{(n-1)}), \qquad n = 1, 2, 3, \dots ,$$

where $\widehat{H}$ denotes a discrete Hamiltonian associated to the corresponding HJ equation. Gauss-Seidel-type or Jacobi-type iterations are possible. For a practical implementation, a criterion of the form

$$\max_{x_i \in G} |\widehat{T}^{(n)}(x_i) - \widehat{T}^{(n-1)}(x_i)| < tol \tag{8}$$

is needed in order to stop the computation at a desired precision *tol*. Clearly this method is neither *local* nor *single-pass*, since the number of iterations needed to reach convergence depends both on the grid size Δx and the dynamics underlying the equation. ITM was proved to be convergent, provided a suitable numerical scheme is employed.

Fast Sweeping Method (FSM) [23, 25] This method is similar to ITM, but the grid is visited in a multiple-direction predefined order. Usually, a rectangular grid is iteratively swept along four directions: $N \to S$, $E \to W$, $S \to N$ and $W \to E$, where N, S, E, W stand for North, South, East and West respectively. This method has been shown to be much faster than ITM, but, as ITM, it is neither *local* nor *single-pass*. A well known exception is given by the eikonal equation, for which it is proved that only 1 sweep (i.e. four visits of the whole grid) is enough to reach convergence (see [25] for details). FSM computes the same solution of ITM, provided the same scheme and the same stopping rule are employed.

Fast Marching Method (FMM) [21, 24] This method has been introduced as a fast solver for the eikonal equation. It differs from the previous ones, since the nodes are visited in a solution-dependent order, producing a *single-pass* method: the algorithm itself finds a correct order for processing the grid nodes. The order which is determined satisfies the *causality* principle, i.e. the computation of a node is declared completed only if its value cannot be affected by the future computation. As recalled in Section 1, at each step the grid is divided in three regions: *ACC*, where computation is definitively done, *CONS*, where computation is going on and *FAR*, where computation is not done yet. Then, the node in *CONS* with the minimal value enters *ACC*, its first neighbours enter *CONS* (if not yet in) and are (re)computed.
Following [22], we remark that this *minimum-value rule* corresponds to compute the value function T step by step in the ascending order (i.e., from the simplex containing $-\nabla T$). It follows that *CONS* expands under the gradient flow of the solution itself, which is exactly equivalent to say that *CONS* is, at each step, an approximation of a level set of the value function. In the case of isotropic eikonal equation (2), the gradient of the solution coincides with the characteristic field of the HJ equation, hence FMM computes the correct solution. Moreover, FMM still works for problems with mild anisotropy, where gradient lines and characteristics define small angles and lie, at each point, in the same simplex of the underlying grid. On the other hand, when a strong anisotropy comes into play, as for a general anisotropic eikonal equation (3), FMM fails and there is no way to compute the viscosity solution following its level sets. Finally, we remark that FMM is also a *local* method, since each node is computed by means of first neighbors nodes only and *CONS* is one-cell thick. Moreover, FMM computes the same solution of ITM, provided the same scheme is employed.

Characteristic Fast Marching Method (CFMM) [5] This method is inspired by FMM, it is *local* and *single-pass* and can be used to solve some eikonal equations. It replaces the search for the minimum value in *CONS* with the search of the node where the characteristic line passes with maximal speed. The acceptance rule is also modified: a node x_i in *CONS* enters *ACC* if the point $x_i + f(x_i, a^*)$ falls in *ACC*. As the Group Marching Method [17], more than one node can enter *ACC* at the same time, making the method in general faster than FMM. Note that CFMM does not always work if the solution of the equation is not differentiable.

Ordered Upwind Method (OUM) [22] This method is inspired by FMM, but it is able to solve more general equations than the eikonal one, including nonhomogeneous anisotropic eikonal equations (4). This can be obtained by enlarging the stencil of the scheme, so that a value at a node x_i can be computed by using values at some nodes x_j that are far from the node x_i. This makes the method *non local*. The maximal allowed distance $|x_i - x_j|$ depends on the degree of anisotropy of the equation. OUM is a *single-pass* method which computes

the same solution of ITM (employing the same numerical scheme) only in the limit $\Delta x \to 0$.
A generalization of OUM has been recently proposed in [1] to solve static convex HJ equations on highly nonuniform grids. The new method MAOUM (Monotone Acceptance OUM) computes the solution in a Fast Marching fashion, but employs large stencils (even larger than OUM) that are pre-computed for each grid node. This makes MAOUM *two-pass* and *non local*.

Buffered Fast Marching Method (BFMM) [8] This method is inspired by FMM and can be used to solve any HJ equations modelling monotone front propagation. Although only first neighbours are involved in the computation, BFMM cannot be considered a *local* method, since *CONS* can increase its thickness. More precisely, the *CONS* region is extended by a *Buffer* region, whose size depends on the dynamics of the equation and, in the worst case, it can cover the whole grid, thus making BFMM comparable to ITM. BFMM is not *single-pass* and computes the same solution of ITM, provided the same scheme is employed.

Progressive Fast Marching Method (PFMM) [3] This method can be considered a localization of BFMM. It is indeed a *local* method, but not *single-pass*. Some experimental results have shown that it can solve quite general problems, including Pursuit-Evasion games with state and control constraints. PFMM has been introduced for theoretical purposes only, since it is very slow (slower than ITM) and then not usable in practice. It proposes a completely new rule for accepting nodes in *CONS*: in the *FAR* region, next to the *CONS* region, a layer of "tempting" values is placed and *progressively* increased, acting as an external boundary condition. For each tempting value, the solution is recomputed in *CONS*, registering the corresponding variations. The first node in *CONS* which is not affected by this external layer enters *ACC*. The "tempting" values can be considered as a guess on the outcome of the future computation and the new rule of acceptance allows one to find the node in *CONS* that cannot be affected by it.

We now consider four additional FM-like methods. In the last two, labelled as *dumb*, the word "methods" should be naively meant. Indeed, they are not new methods for solving HJ equations, rather they are *verification tools*, used to analyze features and limitations of the methods already presented, aiming at giving a comprehensive classification of the equations that can be solved by local and single-pass algorithms. Our ultimate goal is to discuss the possibility that local single-pass methods for solving general HJ equations may not exist. We give two preliminary definitions.

Definition 2.1 (Safe node) *Let $x_i \in$CONS and let $x^*_{i,1}, ..., x^*_{i,p}$ be the neighboring interpolation points of x_i achieving the minimum in* (6) *($p = 2$ or $p = 3$*

depending on the employed SL scheme). Denote by $\lambda^*_{i,1}, ..., \lambda^*_{i,p}$ *the corresponding interpolation weights and define, for* $j = 1, ..., p$, $b_{i,j} = 1$ *if* $x_{i,j} \in ACC$, $b_{i,j} = 0$ *otherwise. The node* x_i *is said to be* safe *if* $\sum_{j=1}^{p} \lambda_{i,j} b_{i,j} = 1$.

The previous definition means that the computation at x_i truly involves values of nodes in *ACC* only. By Remark 2.1, it is clear that the notion of *safeness* makes sense only if the scheme in use employs nodes in *CONS*. Otherwise, all the nodes are safe by construction. Then, we always allow the scheme to use nodes in *CONS*.
From a practical point of view, we remark that round-off errors can prevent the safeness to be satisfied. Then, we can relax this condition by requiring that $\sum_{j=1}^{p} \lambda_{i,j} b_{i,j} \geq 1 - \sigma$ for some tolerance $\sigma > 0$.

Definition 2.2 (Exact node) *A node in CONS is said to be* exact *if its value coincides with that computed by ITM at the same node (only a difference comparable to machine precision is allowed).*

The two new FM-like methods are:

Safe Fast Marching Method (SFMM) This method is identical to FMM but for the rule of acceptance: at each step, the safe node with minimal value in *CONS* enters *ACC*. This method is *local* and *single-pass*.

Safe Method (SM) This method is identical to FMM but for the rule of acceptance: at each step, whichever safe node in *CONS* enters *ACC*. This method is *local* and *single-pass*.

The two verification methods are:

Safe Dumb Method (SDM) This method is identical to FMM but for the rule of acceptance: at each step, whichever safe and exact node in *CONS* enters *ACC*. This method is not usable in practice, since it assumes that one has in hand the solution of ITM (or any other equivalent method). It clearly computes the same solution of ITM and it is *non local*, because it employs the information contained in the final solution of ITM, defined everywhere. SDM is introduced for theoretical purposes, since it represents a kind of upper bound for any *local single-pass* method. This means that if SDM fails, then any *local single-pass* method will fail. Indeed, if there is no safe and exact node in *CONS*, we can conclude that every exact node in *CONS* depends on other nodes in *CONS*. Then, since values at nodes in *CONS* are not "final", there is no way to resolve this loop dependency by keeping the method *local*.

Dumb Method (DM) This method is identical to FMM but for the rule of acceptance: at each step, whichever exact node in *CONS* enters *ACC*. Similar considerations made for SDM apply. This method is introduced for theoretical purposes, since it represents a kind of upper bound for any *single-pass* method. This means that if DM fails, any *single-pass* scheme will fail. The reason is the following: in the limit $\Delta x \to 0$, DM is expected to work for any equation, since, starting from $\mathcal{T}$, at least one characteristic line must cross *CONS* and then at least one exact node must exist in *CONS*. Unexpectedly, for some particular dynamics and choice of Δx, it can happen that no exact node exists in *CONS*. Indeed, the employed grid and scheme can let *flow back* some information from the *FAR* region, so that one has to enlarge the *CONS* region (how large depends again on the dynamics and Δx) and re-compute the solution in order to find one exact node, thus breaking the single-pass property and coming back, in the worst case, to a full grid iterative method like ITM. We stress again that the lack of exact nodes in *CONS* occurs only in pathological cases (see Section 4 for an example), which are solved when $\Delta x \to 0$.

3 Can local single-pass methods solve general HJ equations?

In this section we address the problem of extending the range of applicability of local single-pass methods to general HJ equations. To this end, we focus on the algorithms discussed in the previous section which are local and single-pass, namely FMM, SFMM and SM. In order to point out their features and limitations, we will also employ the two verification methods SDM and DM.

Since the introduction of FMM, HJ equations have been divided in two classes. Given a mesh, we have:

(EIK) Eikonal-like equations, whose characteristic lines coincide or lie in the same simplex of the gradient lines of their solutions.

(¬EIK) Non eikonal-like equations, for which there exists at least a grid node where the characteristic line and the gradient of the solution do not lie in the same simplex.

FMM works for equations of type EIK and fails for equations of type ¬EIK (see [22] for further details and explanations). Let us introduce two other classes for HJ equations of type (5):

(DIFF) Equations with smooth characteristics. Information spreads from the target $\mathcal{T}$ to the rest of the space along smooth lines, without shocks. The

solution T is differentiable.

(¬DIFF) Equations with nonsmooth characteristics. Information starts from the target $\mathcal{T}$ and then crashes, creating shocks. The solution T is Lipschitz continuous.

Let us comment the applicability of the local single-pass methods by making use of the classifications introduced above.

(1) SM solves DIFF. SM can be applied in the case DIFF, provided SDM works. Let us denote by x_i one safe node in *CONS* (x_i exists because we assume SDM can be applied). By definition of safeness, the value at x_i only depends on values at nodes in *ACC*, that can be assumed to be exact by induction. Then, the exactness of the value at x_i is guaranteed by the property DIFF, which implies that no characteristics will reach x_i in the future from another direction, possibly changing its value. In other words, the information passes through x_i one and only one time. Then, once x_i is reached by the region *ACC*, it is ready to enter *ACC*.

(2) Is the minimum-value rule really needed? Having in mind the FMM (and its ancestor, the Dijkstra's algorithm [11]), one can be convinced that giving priority to the smallest value among nodes in *CONS* is an essential request to make the method work. On the contrary, by the above comment (1), we know that a method like SM, which makes no distinction among nodes with respect to their values, works in the case DIFF (both EIK and ¬EIK), provided SDM works. The choice of the minimum value becomes essential only in the ¬DIFF case, where characteristics reach some point from two or more different directions. We discuss this point in the next comment.

(3) Handling shocks in the ¬DIFF case. Let us consider the ¬DIFF case and let x be a point belonging to a shock, i.e. where the solution is not differentiable. By definition, the value $T(x)$ is carried by two or more characteristic lines reaching x at the same time. Similarly, let x_i be a grid node Δx-close to the shock. In order to mimic the continuous case, x_i has to be approached by the *ACC* region approximately at the same time from the directions corresponding to the characteristic lines. In this case, the value $T(x_i)$ is correct (no matter the upwind direction is chosen) and, more important, the characteristic information stops at x_i and it is no longer propagated, getting stuck by the *ACC* region. As a consequence, the shock is localized properly.
It is interesting to note that for EIK equations, this property is satisfied by FMM that, thanks to the minimum-value rule, guarantees the evolving region *CONS* to be, at any time, a good approximation of the level sets of the final solution.

(4) ¬EIK case requires CONS not to be an approximation of the level sets of the solution. In order to solve correctly ¬EIK equations, *CONS* cannot be at any time an approximation of the level sets of the solution. This is due to the fact that the anisotropy shifts the characteristic directions, so that they no more coincide with the gradient directions. *CONS*, to be correctly enlarged, should not follow the gradient direction and then it coincides no more with the level sets. In Section 4 we show an example for equation (3). See also [22] for a more detailed explanation.

(5) Requiem for local single-pass methods solving general HJ equations? Let us consider the ¬EIK & ¬DIFF case. By comments (3)-(4), in order to solve ¬EIK equations, the *CONS* region cannot be an approximation of a level set of the solution. But, doing so, a node x_i close to a shock can be reached by ACC at different times. When ACC reaches x_i for the first time, it is impossible to detect the presence of the shock by using only local information. Indeed, only a global view of the solution allows one to know that another characteristic line will reach x_i at a later time. As a consequence, the algorithm continues the enlargement of *CONS* and ACC, thus making an error that cannot be redressed in the future. Test 4 in Section 4 shows an example in which a shock crosses a region with strong anisotropy. In this situation, it seems impossible to get the correct solution without the addition of nonlocal information regarding the location of the shock, or going back to nodes in ACC at later time.

Table 1 summarizes the comments above. Note that the word "no" in the table should be meant as "not in general", since some exceptions are possible. It is plain that SFMM is the more versatile among all methods (whenever it can be applied), since it joins advantages of both SM and FMM.

Table 1: A bird-eye view on the applicability of local single-pass methods

	EIK & DIFF	EIK & ¬DIFF	¬EIK & DIFF	¬EIK & ¬DIFF
FMM	yes	yes	no	no
SM	yes	no	yes (if SDM works)	no
SFMM	yes	yes	yes (if SDM works)	no

4 Numerical tests

The first aim of this section is comparing the two semi-Lagrangian schemes described in Sections 2.1.1-2.1.2. In the following we will denote the two schemes by SL-2p and SL-3p, respectively. The second aim is confirming the theoretical observations in Section 3, and investigating the theoretical bounds given by SDM and DM, in order to understand how much SFMM is close to that limit. This

will give an idea about how much room is still present for further improvements in the field of local single-pass methods. In all the tests the solution computed by FSM on a fine 801^2 grid will be referred to as the "exact" solution and will be denoted by T^{exact}.

In the following we list five reference HJB equations, together with the class they belong to (see Table 2). In all the cases we set $d = 2$, $A = B(0,1)$, $a = (a_1, a_2) \in A$ and $\mathcal{T} = \{(0,0)\}$. Moreover, λ, μ and ε denote generic positive parameters. Finally we define $m_{\lambda,\mu}(a) = (1 + (\lambda\, a_1 + \mu\, a_2)^2)^{-\frac{1}{2}}$ and we denote by χ_S the characteristic function of a set S.

Table 2: Equations considered for numerical tests and the classes they belong to

Equation	Dynamics	Class
HJB-A	$f(x,y,a) = a$	EIK & DIFF
HJB-B	$f(x,y,a) = (1 + \chi_{\{x>1\}})\, a$	EIK & ¬DIFF
HJB-C	$f(x,y,a) = m_{\lambda,\mu}(a)\, a$	¬EIK & DIFF
HJB-D	$f(x,y,a) = (m_{\lambda,\mu}(a) + \varepsilon(x-1)\chi_{\{x>1\}})\, a$	¬EIK & ¬DIFF
HJB-E	$f(x,y,a) = (1 + \lvert x+y\rvert) m_{\lambda,\mu}(a)\, a$	¬EIK & ¬DIFF

Test 0 (SL-2p vs SL-3p). In this test we compare the schemes described in Sections 2.1.1-2.1.2 by means of FSM, in terms of accuracy and number of iterations. We consider equations HJB-A and HJB-D (for $\varepsilon = 0.02$). Relative errors in norm L^1 and L^∞ with respect to the "exact" solution T^{exact} are defined as

$$E_1 := \frac{1}{N}\sum_{i=1}^{N} \frac{|T^{exact}(x_i) - \widehat{T}(x_i)|}{|T^{exact}(x_i)|} \qquad \text{and} \qquad E_\infty := \max_{i=1,\dots,N} \frac{|T^{exact}(x_i) - \widehat{T}(x_i)|}{|T^{exact}(x_i)|}\,.$$

By "sweep" we mean *four* iterations executed in four different directions. When reporting the number of sweeps of FSM, we include the final "stopping" sweep, needed to realize that convergence is reached, namely the stopping rule (8) is satisfied. We choose *tol* =1e-16 (machine precision). Results are reported in Table 3.

We recall that, as discussed in Section 2.2, the convergence of FSM is ensured in 1 sweep for equation HJB-A, see [25]. Nevertheless, real algorithms involving double precision computations can require 2 sweeps to reach the machine precision. The third sweep reported in Table 3 is the "stopping" sweep. It is rather clear that SL-3p overcomes SL-2p in terms of both accuracy and number of sweeps. This is likely due to the fact that SL-3p can propagate the characteristic information of the HJ equation along diagonal directions easier than SL-2p. From now on, only the scheme SL-3p will be employed for all the following tests.

Test 1 (EIK & DIFF). In this test we compare FSM, FMM and SM against HJB-A. Errors with respect to the "exact" solution T^{exact} are reported

Table 3: Test 0: SL-2p and SL-3p schemes comparison

equation	grid	scheme	E_∞	E_1	# sweeps
HJB-A	101^2	SL-2p	0.130	0.016	3
HJB-A	101^2	SL-3p	0.079	0.009	3
HJB-A	201^2	SL-2p	0.094	0.008	3
HJB-A	201^2	SL-3p	0.058	0.004	3
HJB-A	401^2	SL-2p	0.050	0.003	3
HJB-A	401^2	SL-3p	0.030	0.002	3
HJB-D	101^2	SL-2p	0.888	0.053	8
HJB-D	101^2	SL-3p	0.635	0.029	4
HJB-D	201^2	SL-2p	0.535	0.027	7
HJB-D	201^2	SL-3p	0.405	0.014	4
HJB-D	401^2	SL-2p	0.245	0.010	7
HJB-D	401^2	SL-3p	0.189	0.005	3

in Table 4.

Table 4: Test 1: EIK & DIFF

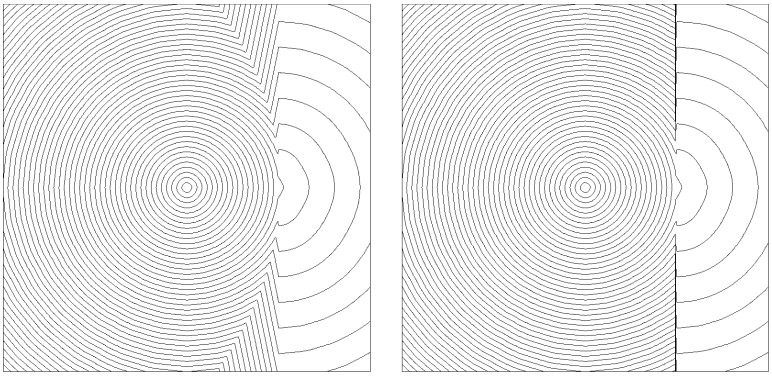
grid	method	E_∞	E_1
101^2	FSM	0.079	0.009
101^2	FMM	0.079	0.009
101^2	SM	0.079	0.009
201^2	FSM	0.057	0.004
201^2	FMM	0.057	0.004
201^2	SM	0.057	0.004
401^2	FSM	0.029	0.001
401^2	FMM	0.029	0.001
401^2	SM	0.029	0.001

The three methods lead to the same error because they compute exactly the same solution. A fortiori, also SFMM does. This confirms that SM can be applied in EIK & DIFF cases and that picking the minimum value in *CONS*, as acceptance rule, is not strictly needed here to compute the correct solution.

Test 2 (EIK & ¬DIFF). In this test we compare FSM, FMM, SFMM and SM against HJB-B. Errors with respect to the "exact" solution T^{exact} are reported in Table 5. FSM, FMM and SFMM lead to the same error because they compute exactly the same solution. Conversely, SM cannot be used here, since it is not able to properly locate the shocks (see Fig. 2).

Table 5: Test 2: EIK & ¬DIFF

grid	method	E_∞	E_1
101^2	FSM	0.079	0.011
101^2	FMM	0.079	0.011
101^2	SFMM	0.079	0.011
101^2	SM	0.583	0.019
201^2	FSM	0.057	0.006
201^2	FMM	0.057	0.006
201^2	SFMM	0.057	0.006
201^2	SM	0.606	0.014
401^2	FSM	0.029	0.002
401^2	FMM	0.029	0.002
401^2	SFMM	0.029	0.002
401^2	SM	0.603	0.011



FSM SM

Figure 2: Test 2: level sets of the solutions computed by FSM and SM

Test 3 (¬EIK & DIFF). In this test we compare FSM, FMM and SM against HJB-C. Errors with respect to the "exact" solution T^{exact} are reported in Table 6. FSM and SM lead to the same error because they compute exactly the same solution. A fortiori, also SFMM does. Conversely, FMM fails (although it is quite robust), since it does not compute the same solution of FSM. This comes from the fact that FMM is not able to deal with substantial anisotropies, as discussed in Section 2.2 (see also [22] for more details).

Test 4 (¬EIK & ¬DIFF). In this test we compare FSM, FMM, SFMM, SM, DM and SDM against HJB-E (for $\lambda = 6$ and $\mu = 5$). Fig. 3 shows some optimal directions (characteristic lines) computed by means of FSM.

Table 6: Test 3: ¬EIK & DIFF

grid	method	E_∞	E_1
101^2	FSM	0.635	0.029
101^2	FMM	0.635	0.058
101^2	SM	0.635	0.029
201^2	FSM	0.404	0.014
201^2	FMM	0.408	0.049
201^2	SM	0.404	0.014
401^2	FSM	0.189	0.005
401^2	FMM	0.290	0.044
401^2	SM	0.189	0.005

Both the strong inhomogeneity (characteristic lines bend hardly in the I and III quadrant) and the shock (the cubic-like curve in the II and IV quadrant) are visible.

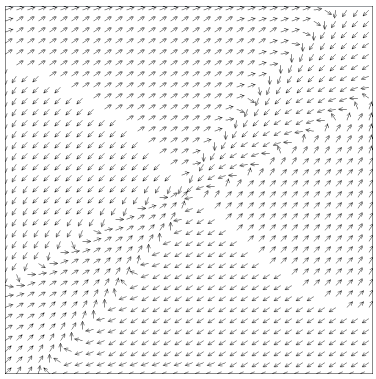


Figure 3: Test 4: Optimal vector field f^*_{FSM} computed by FSM (zoom around the origin)

Table 7 reports the error E_1 with respect to the "exact" solution T^{exact}. In this case only "dumb" methods (DM and SDM) are able to compute the same solution of FSM, although SFMM is very close to FSM. The differences among the methods are much more evident looking at the level sets of the corresponding solutions, reported in Fig. 4. FSM is able to respect the anisotropy, indeed the level sets of its solution around the origin are ellipses as expected. Moreover, it properly catches the shock. FMM tries catching the shock, but fails in respecting the anisotropy. SM tries respecting anisotropy, but fails in catching the shock. Finally, SFMM is a kind of mix between FMM and SM.

Table 7: Test 4: ¬EIK & ¬DIFF

grid/method	FSM	FMM	SFMM	SM	DM	SDM
101^2	0.114	0.170	0.115	0.124	0.114	0.114
201^2	0.061	0.132	0.062	0.072	0.061	0.061
401^2	0.024	0.109	0.025	0.036	0.024	0.024

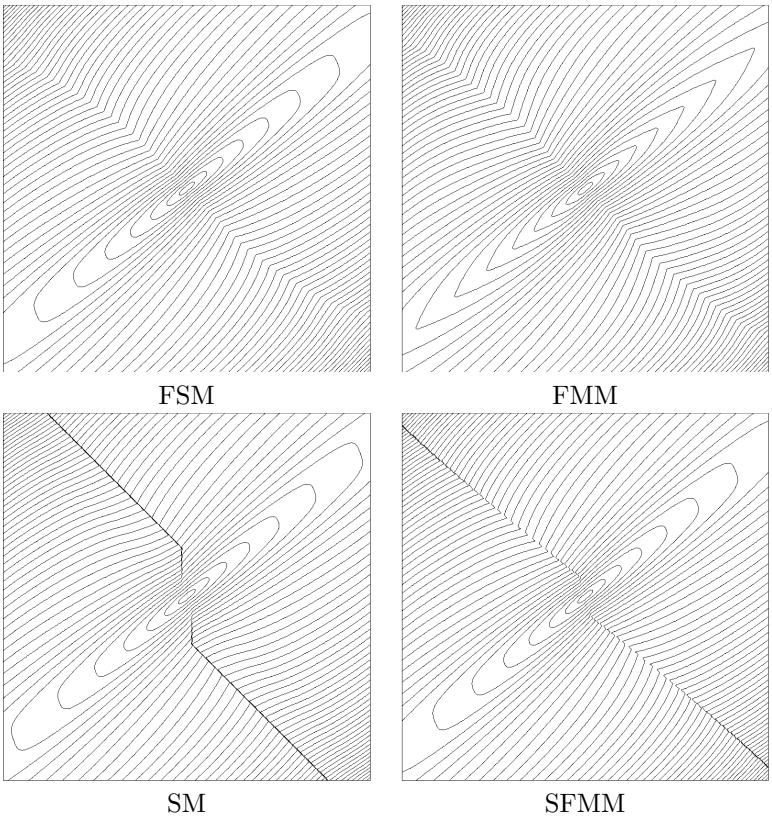
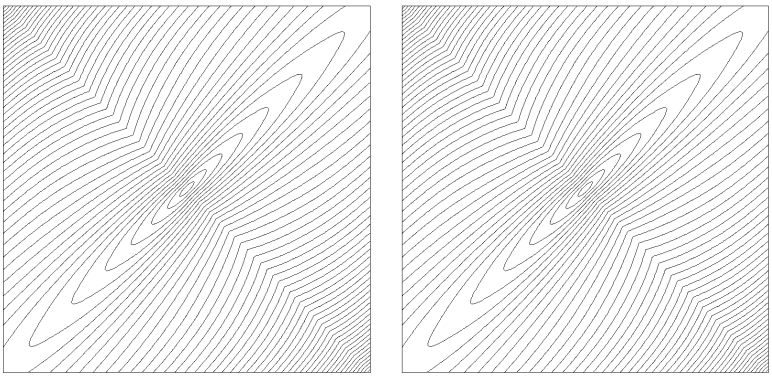


Figure 4: Test 4: level sets of the solutions computed by FSM, FMM, SM and SFMM

Test 5 (¬EIK & ¬DIFF - easy case). In this test we compare FSM and SFMM against HJB-E (for $\lambda = 5$ and $\mu = 5$). Due to the fact that $\lambda = \mu$, the shock has a particular symmetry with respect to the axes. This symmetry makes SFMM work, since *CONS* "luckily" reaches the shock at the same time from both sides (see Fig. 5). This example shows that local single-pass schemes can solve ¬EIK & ¬DIFF equations in some special cases.



FSM SFMM

Figure 5: Test 5: level sets of the solutions computed by FSM and SFMM

Test 6 (¬EIK & ¬DIFF - hard case). In this test we show that even SDM and DM can fail in computing the correct solution, i.e. it can happen that either there is no safe node in *CONS* and/or there is no exact node in *CONS*. Therefore, the methods stop abruptly before *ACC* covers the whole domain. We consider again the equation HJB-E (for $\lambda = 10$ and $\mu = 5$). This case is even more pathological than that depicted in Fig. 3: characteristic lines bend too much compared to the mesh size, i.e. they can significantly change direction within a single cell. We caught the precise moment in which both SDM and DM stop working, due to the lack of safe and exact nodes in *CONS*. In Fig. 6 the black central node is the target, gray nodes represent the *ACC* region, whereas white nodes are in *CONS*. For each node in *CONS* we plot the optimal vector field f^*_{SDM} computed by means of the current solution of SDM. It is evident that every node in *CONS* depends on other nodes in *CONS*, so that a loop is created and no safe node is present.

In Fig. 7 we show a detail of Fig. 6 and we plot the optimal vector fields f^*_{DM} (in black) and f^*_{ITM} (in red), computed by means of the solution of DM and ITM respectively (if the two optimal vector fields coincide only one is plotted).

It is easy to see that f^*_{ITM} points either toward the *FAR* region or toward nodes in *CONS*, whose f^*_{ITM} also points toward the *FAR* region (recursively). This means that the values at nodes in *ACC* and *CONS* are not enough to compute exact values in *CONS*, even if we perform an additional stabilization by iterating the scheme on *CONS* up to convergence. This is also confirmed by the fact that ITM requires in this case a huge number of iterations to reach convergence, compared to that of the previous tests. The thickness of *CONS* must be increased (as for BFMM), in order to solve the dependency. Although the lack of exact nodes should completely disappear for $\Delta x \to 0$ and $N_c \to +\infty$, this test shows that on finite grids some pathological cases can arise.

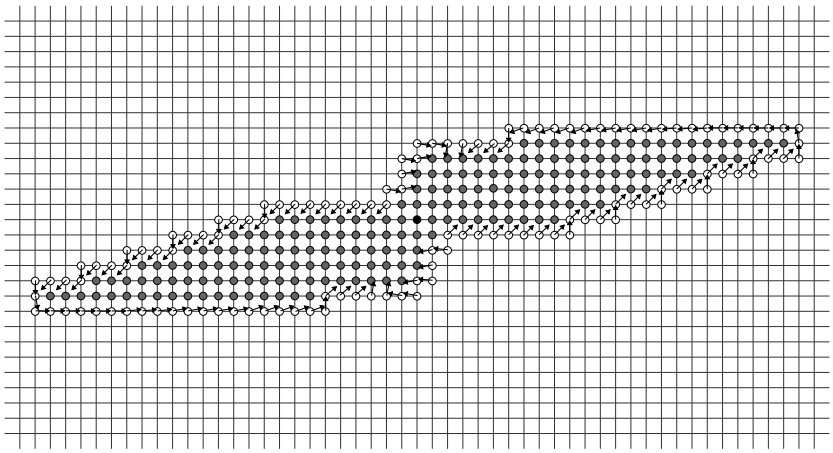


Figure 6: Test 6: SDM fail because no safe nodes are found in *CONS*. Note the loop dependency following the optimal vector field f^*_{SDM}

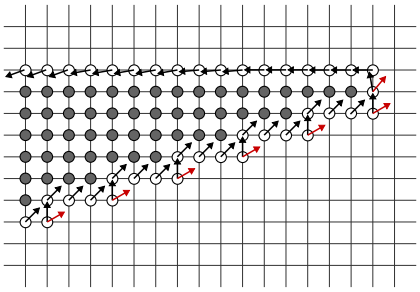


Figure 7: Test 6: DM fail because no exact nodes are found in *CONS*. Exact information flows back from *FAR* region breaking locality and single-pass property

5 Conclusions

The above tests and considerations allow us to sketch some final comments.

1. We want to stress that all the considerations debated in the paper have a theoretical value. Indeed, from the practical point of view, it is not always possible to know in advance if an equation falls in the class EIK or DIFF and then it is not evident how to choose a method which is able to solve it. Only ITM and FSM can be safely used if no *a priori* knowledge of the solution is available.

2. Let us add some words about SM. This is one of the simplest methods one can imagine, nevertheless it is able to solve a large class of equations, including the homogeneous anisotropic eikonal equation (3). As a consequence, methods

such OUM and PFMM are in some sense "too complicated" than necessary. In our opinion, the reason why the minimum-value rule was considered as crucial so far is simply that Dijkstra method uses it. But, on graphs, the distinction between DIFF and ¬DIFF is not visible, as well as the condition of safeness. Moreover, the importance of the safeness was missed because the distinction in the usage of *CONS* nodes described in Remark 2.1 was completely underestimated. In this respect, let us point out that SM is very similar to CFMM. Indeed, the acceptance rule used in CFMM actually coincides with that of SM, cf. Definition 2.1. In [5] it was already noted that, running CFMM, *CONS* does not coincide with the level set of the solution, but this fact was not fully exploited as we do in this paper.

3. The reliability of the SFMM against ¬EIK & ¬DIFF equations cannot be known in advance. If the method computes a solution, i.e. ACC covers the whole grid, that solution can be correct or not. Otherwise, if the method stops, due to the lack of safe nodes in *CONS*, the user can definitively conclude that this method cannot be used for the equation at hand.

4. Our experience suggests that there is no much room between SFMM and SDM, meaning that it is quite difficult to precisely define a class of equations that can be solved by SDM and not by SFMM. Since we have seen that SDM is a sort of a limit for local single-pass schemes, we shall conclude that it is relatively fruitless investigating new local single-pass schemes.

Acknowledgements

Authors are grateful to Alexander Vladimirsky and Michael Breuß for the interesting and motivating discussions they had with them during the preparation of this paper.

References

[1] K. Alton and I. M. Mitchell, *An ordered upwind method with precomputed stencil and monotone node acceptance for solving static convex Hamilton-Jacobi equations*, J. Sci. Comput., 51 (2012), pp. 313–348.

[2] M. Bardi and I. Capuzzo Dolcetta, *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*, Birkhäuser, Boston, 1997.

[3] S. Cacace, E. Cristiani, and M. Falcone, *A local ordered upwind method for Hamilton-Jacobi and Isaacs equations*, Proceedings of 18th IFAC World Congress 2011, pp. 6800–6805.

[4] D. L. Chopp, *Some improvements of the fast marching method*, SIAM J. Sci. Comput., 23 (2001), pp. 230–244.

[5] E. Cristiani and M. Falcone, *A characteristics driven Fast Marching method for the eikonal equation*, in K. Kunisch, G. Of, O. Steinbach (Eds.), Numerical Mathematics and Advanced Applications (ENUMATH 2007, Graz, Austria, September 10-14, 2007), pp. 695–702, Springer, Berlin Heidelberg, 2008.

[6] E. Carlini, M. Falcone, N. Forcadel, and R. Monneau, *Convergence of a Generalized Fast Marching Method for an Eikonal equation with a velocity changing sign*, SIAM J. Numer. Anal., 46 (2008), pp. 2920–2952.

[7] A. Chacon and A. Vladimirsky, *Fast two-scale methods for eikonal equations*, SIAM J. Sci. Comput., 34 (2012), pp. 547–578.

[8] E. Cristiani, *A fast marching method for Hamilton-Jacobi equations modeling monotone front propagations*, J. Sci. Comput., 39 (2009), pp. 189–205.

[9] E. Cristiani and M. Falcone, *Fast semi-Lagrangian schemes for the Eikonal equation and applications*, SIAM J. Numer. Anal., 45 (2007), pp. 1979–2011.

[10] P. Danielsson, *Euclidean distance mapping*, Computer Graphics and Image Processing, 14 (1980), pp. 227–248.

[11] E. W. Dijkstra, *A note on two problems in connexion with graphs*, Numer. Math., 1 (1959), pp. 269–271.

[12] M. Falcone, *The minimum time problem and its applications to front propagation*, in A. Visintin e G. Buttazzo (Eds.), "Motion by mean curvature and related topics", De Gruyter Verlag, Berlin, 1994.

[13] M. Falcone, *Numerical solution of dynamic programming equations*, Appendix A in [2].

[14] J.J. Helmsen, E.G. Puckett, P. Colella, and M. Dorr, *Two new methods for simulating photolithography development in 3d*, Proc. SPIE 2726, Optical Microlithography IX (1996), pp. 253–261.

[15] W.-K. Jeong, R. T. Whitaker, *A Fast Iterative Method for Eikonal Equations*, SIAM J. Sci. Comput., 30 (2008), pp. 2512–2534.

[16] C. Y. Kao , S. Osher , J. Qian, *Lax–Friedrichs sweeping scheme for static Hamilton–Jacobi equations*, J. Comput. Phys., 196 (2004), pp. 367–391.

[17] S. Kim, *An $O(N)$ level set method for eikonal equations*, SIAM J. Sci. Comput., 22 (2001), pp. 2178–2193.

[18] W. M. McEneaney, *Max-Plus Methods for Nonlinear Control and Estimation*, Birkhäuser Systems and Control Series, 2006.

[19] E. Prados and S. Soatto, *Fast Marching Method for Generic Shape from Shading*, N. Paragios et al. (Eds.), VLSM 2005, Lecture Notes in Computer Science 3752, pp. 320–331, Springer-Verlag Berlin Heidelberg, 2005.

[20] E. Rouy and A. Tourin, *A viscosity solutions approach to Shape-from-Shading* SIAM J. Numer. Anal., 29 (1992), pp. 867–884.

[21] J. A. Sethian, *A fast marching level set method for monotonically advancing fronts*, Proc. Natl. Acad. Sci. USA, 93 (1996), pp. 1591–1595.

[22] J. A. Sethian and A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations: theory and algorithms*, SIAM J. Numer. Anal., 41 (2003), pp. 325–363.

[23] Y. Tsai, L. Cheng, S. Osher and H. Zhao, *Fast sweeping algorithms for a class of Hamilton-Jacobi equations*, SIAM J. Numer. Anal., 41 (2004), pp. 673–694.

[24] J. N. Tsitsiklis, *Efficient algorithms for globally optimal trajectories*, IEEE Trans. Automat. Control, 40 (1995), pp. 1528–1538.

[25] H. Zhao, *A fast sweeping method for Eikonal equations*, Math. Comp., 74 (2005), pp. 603–627.